CASPER: Using Local Search for Planning for Embedded Systems

Steve Chien, Gregg Rabideau, Russell Knight

Jet Propulsion Laboratory
California Institute of Technology

For further information see: casper.jpl.nasa.gov

ABSTRACT

The Continuous Activity Scheduling Planning, Execution and Replanning (CASPER) system has been used for a wide range of automation applications ranging from spacecraft, rovers, ground communications stations, to unpiloted aerial vehicles. In almost all of these applications, CASPER uses at its core a local search algorithm to rapidly solve planning and scheduling problems. In this paper we describe the basics of this local search algorithm as well as outline several of the benefits and drawbacks of using a local search approach.

INTRODUCTION

In the past few years, exciting robotic missions such as Galileo, Clementine, Mars Pathfinder, Lunar Prospector, and Cassini have opened new vistas in space exploration. While each of these missions contains some automation, such ambitious missions still require large teams of experts working around the clock to generate and validate spacecraft command sequences. Increasing knowledge of our Earth, our planetary system, and our universe challenges NASA to fly numerous ambitious missions; fiscal realities require that NASA do so with ever-smaller budgets. Given this, the automation of spacecraft commanding becomes an endeavor of crucial importance.

Automated planning and scheduling technology is one aspect of spacecraft commanding that offers great promise in enabling autonomous spacecraft. Such a spacecraft is goal-based: it knows its goals and automatically selects and performs activities to achieve them. Goals might include science goals, such as "perform a mapping campaign using the ultraviolet spectrometer," and engineering goals, such as "maintain propulsion-system health." Because planning and scheduling technology allows goal-based commanding, it can dramatically reduce operations costs and onboard response times to system faults or science opportunities.

Automated planning and scheduling technology is applicable to a wide spectrum of missions, from those with limited onboard computational capabilities, such as Lunar Prospector, to those with highly sophisticated software, such as Cassini. In all cases, the goal is for the mission scientist to command the spacecraft directly, with no need for mission operations specialists to perform routine activities. In the most sophisticated missions, the spacecraft operates autonomously, only interacting with the ground systems and personnel to schedule a downlink activity to transmit science data back to Earth. These autonomous spacecraft contain complex onboard software that provides knowledge and reasoning procedures to track goals, spacecraft resources, and spacecraft state and deliberately plan activities to achieve goals while respecting spacecraft operations constraints.

In this paper, we focus on the CASPER automated planning system, which is being used in a range of space operations applciations. Specifically, we describe the local search framework most commonly used within the CASPER continuous planning system. We begin by describing the basic representational framework underlying CASPER. We then give a brief overview of the iterative repair algorithms. Next we describe the algorithm portfolio framework used to enable escape from local maxima. Finally, we describe ongoing deployments of the CASPER system and related work.

REPRESENTATIONAL FRAMEWORK

We consider a very general class of planning and scheduling problems incorporating states, resources, functional dependencies, metric time, and hierarchical activities [Sherwood et al., 1998]. We now describe this representational framework. We define a *parameter* as a variable with a restricted domain (e.g., the range of integers between ten and twenty, floating point numbers, booleans and strings). A *parameter dependency* is a functional relationship between two parameters. An activity end time, for example, is a function (the sum) of the start time and the duration. A more complicated dependency might compute the duration of a spacecraft slew from the initial and final orientation.

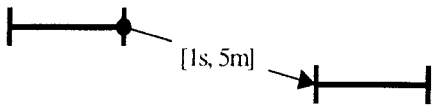


Figure 1: A temporal constraint with a required separation of at least 1 second and at most 5 minutes.

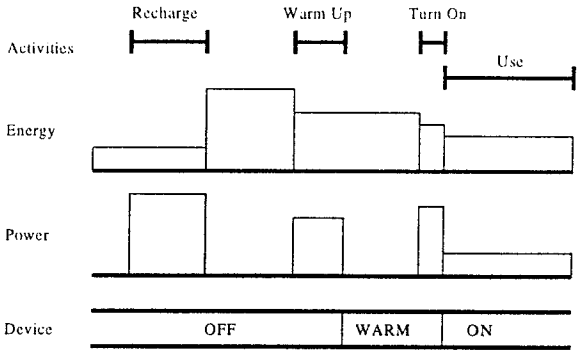


Figure 2: Timelines for activities, a depletable resource (energy), a non-depletable resource (power), and a state variable (device).

In the model, relative ordering constraints can be specified for pairs of activities. A *temporal constraint* is a relationship between the start or end time of one activity with the start or end time of another activity (see Figure 1). For example, an instrument might require calibration between 1s and 5 minutes before an observation. Minimum and maximum separation distances can be specified in a temporal constraint. The warming activity for example, might be required to end at least one second but at most five minutes before using the instrument. Temporal constraints can be combined with conjunctive or disjunctive operators to form more complicated expressions.

A *resource* represents the profile of a physical resource or system variable over time (see Figure 2), as well as the upper and lower bounds of the profile. In ASPEN, a resource can either be depletable or non-depletable. A depletable resource is used by a reservation and remains used even after the end of the activity making the reservation. Examples of depletable resources on spacecraft include memory, fuel and energy. A non-depletable resource is used only for the duration of the activity making the reservation. Solar power is an example of a non-depletable resource. A resource can be assigned a capacity, restricting its value at any given time. A *state variable* represents the value of a discrete system variable over time. The set of possible states and the set of allowable transitions between states are both defined with the state variable. An example of a state variable is an instrument switch that may be either ON, WARMING, or OFF. This state variable may be restricted to transitions from OFF to WARMING but not directly to ON. *Reservations* are requirements of activities on resources or state variables. For example, an activity can have a reservation for ten watts of power. Some reservations are modeled as instantaneous effects (e.g., reservations that change the state on a state variable). The user can specify whether this effect occurs at the start or end of the activity.
Activity hierarchies can be specified in the model using decompositions, a set of sub-activities along with temporal constraints between them. In this way, one can define a high-level activity that decomposes into a set of lower-level activities that may be required to occur in some relative order. These activities in turn may have their own decompositions. In addition, an activity may have multiple decompositions to choose from. Thus, allowing an activity to be expanded in different ways. An *activity* has a set of parameters, parameter dependencies, temporal constraints, reservations and decompositions. All activities have at least three parameters: a start time, an end time, and duration.

Plan Conflicts and Repair

While there are several planning and scheduling search strategies implemented in ASPEN, the most commonly used algorithm is called *iterative repair* [Zweben et al., 1994]. During iterative repair, the conflicts in the schedule are detected and addressed one at a time until no conflicts exist, or a user-defined time limit has been exceeded. We define a conflict as a particular class of ways to violate a plan constraint (e.g., over-use of a resource or an illegal state transition). For each conflict type, there is a set of repair methods. The search space consists of all possible repair methods applied to all possible conflicts in all possible orders. We describe an efficient approach to searching this space. Conflicts can be repaired by means of several predefined methods. The repair methods are: moving an activity, adding a new instance of an activity, deleting an activity, detailing an activity, abstracting an activity, making a reservation of an activity, canceling a reservation, connecting a temporal constraint, disconnecting a constraint, and changing a parameter value. The repair algorithm first selects a conflict to repair then selects a repair method. The type of conflict being resolved determines which methods can repair the conflict. Depending on the selected method, the algorithm may need to make addition decisions. For example, when moving an activity, the algorithm must select a new start time for the activity.

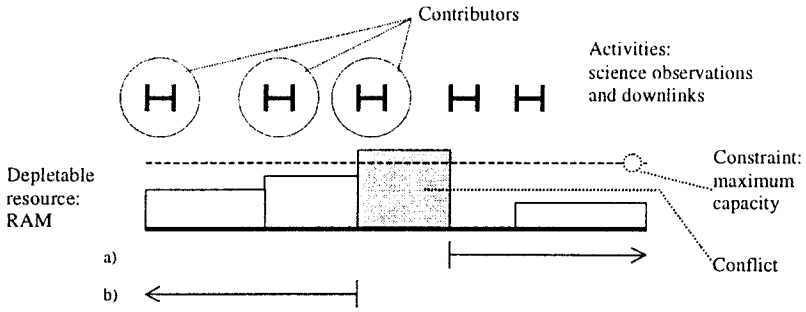


Figure 3: Repairing a depletable resource conflict. The arrows show time intervals that resolve the conflict by a) moving a positive contributor or b) adding a negative contributor.

Figure 3 shows an example situation for repair. On-board RAM is represented as a depletable resource. The shaded region shows a conflict where the RAM buffer has been oversubscribed. The science activities using the resource prior to the conflict are considered contributors. Moving or deleting one of the contributors can repair the conflict. Another possibility would be to create a new downlink activity in order to replenish the resource and repair the conflict.

Continuous Planning

Onboard planning enables integration of the planning process with execution to provide feedback. However, in an onboard planning context, the planner is an embedded entity that makes the batch-oriented model of planning inappropriate. Specifically, such an embedded planner must be anytime and responsive. It must be anytime in that at any point in time there must be an executable plan. This means that generative planning techniques [Jonsson et al 2000] are less suitable because they do not have the "anytime" property.

In order to address this issue, the proposers have developed the Continuous Activity Scheduling Planning Execution and Replanning (CASPER) [Chien et al. 2000a] system - a soft, real-time version of the ASPEN planning system. Rather than considering planning a batch process in which a planner it is presented with goals and an initial state (see Figure 4), CASPER has a current goal set, a current state and projections into the future, and a current plan. At any time an incremental update to the goals or current state (an unexpected event or simply time progressing forward) may update the planner process (see Figure 5). The planner is then responsible for maintaining a consistent, satisficing plan for the most current information. Incremental changes to the goals, initial state, or executed activities trigger iterative repair conflicts with the plan.

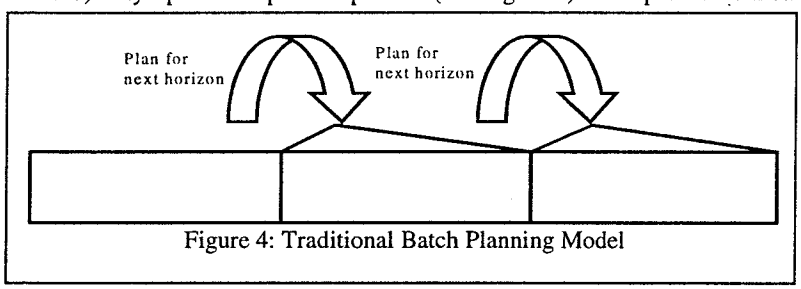


Figure 4: Traditional Batch Planning Model

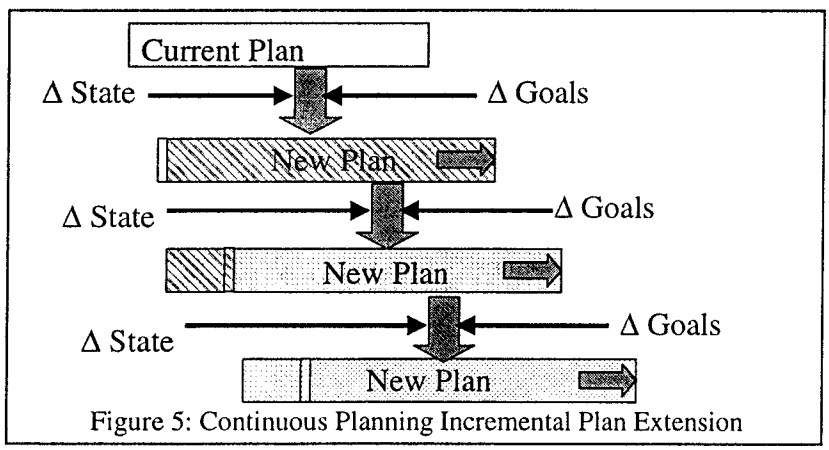


Figure 5: Continuous Planning Incremental Plan Extension

CASPERs design goal is to respond to activity and state updates on 1-10 second timescale. This enables more up to date information regarding the execution status of activities as well as monitored state and resource values to influence planning. The benefit of this responsiveness can be motivated Space Infra-red Telescope Facility (SIRTF) operations scenarios. In this operations scenario, the observatory is in a near earth orbit and has a set of observation targets. However, it is difficult to project exactly how future execution of the plan will proceed. For example, if spacecraft is able to acquire the target quickly (as compared to conservative settling times and time

for search for the target), an observation may complete significantly ahead of schedule. Alternatively, if the spacecraft repeatedly fails to acquire a guidestar required by an observation, an observation may be terminated. This also has the effect of completing the activity ahead of schedule but with a failed outcome. Within this operations context, a short-term planner would decide which observations to sequence next. Such a planner would need to consider all targets currently on the observation list, their visibility windows, and their relative positions in the sky (for reasons of slew minimization and for observation quality issues). The short term planner would also need to track other resource management issues such as data management relating to engineering and science observations and coordination with downlink windows.

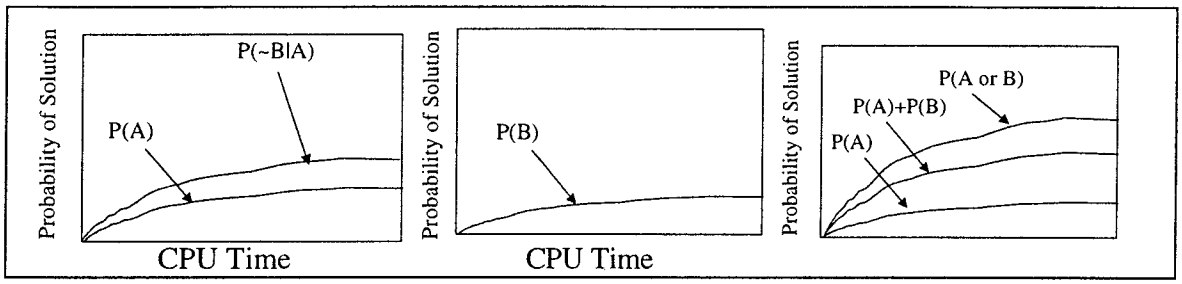
Search in ASPEN has focused on high speed, local search, in a committed plan space, using stochastic combination of a portfolio [Huberman et al. 1997, Gomes and Selman 1997] of heuristics for iterative repair and improvement algorithms. In this approach, at each choice point in the iterative repair process [Rabideau, et al. 1999], a stochastic choice is made among a portfolio of heuristics (with probabilities specifiable by the user). This approach has performed well in a wide range of space mission applications [Chien et al 2000b] including spacecraft operations scheduling, rover planning, ground communications station automation and autonomy for Uninhabited Aerial Vehicles. The stochastic element combined with a portfolio of heuristics helps to avoid the typical pitfalls of local search. Using a committed plan representation enables fast search moves and propagation of effects (100s of operations per CPU second). To increase efficiency, we also make use of aggregates of activities [Knight et al 2000].

We have focused on an early-commitment, local, heuristic, iterative search approach to planning, scheduling and optimization. This approach has a number of desirable properties for spacecraft operations planning.

1. Using an iterative algorithm allows automated planning to be utilized at any time and on any given initial plan. The initial plan may be as incomplete as a set of goals, or it may be a previously produced plan with only a few flaws. Repairing and optimizing an existing plan enables fast replanning when necessary from manual plan modifications or from unexpected differences detected during execution. This enables local search planning to have an *anytime* property, in which it always has a "current best" solution and improves it as time and other resources allow. Refinement search methods [Jonsson et al 2000] do not have this property. Local search can also be used in a "mixed initiative" mode for partial ground-based automation.
2. It is easier to write powerful heuristics that evaluate ground plans. These strong heuristics allow the search to be pruned, ruling out less promising planning choices.
3. A local algorithm does not incur the overhead of maintaining intermediate plans or past attempts. This allows the planner to quickly try many plan modifications for repairing the conflicts or improving the preferences. However, unlike systematic search algorithms, it cannot be guaranteed that our iterative algorithms will explore all possible combinations of plan modifications or that it will not retry unhelpful modifications. In our experience, these guarantees are not valuable because for large-scale problems complete search is intractable.
4. By committing to values for parameters, such as activity start times and resource usages, the effects of a resource usage and the corresponding resource profiles can be efficiently computed. Least-commitment techniques retain plan flexibility, but can be computationally expensive for large applications. Further discussions on this topic can be found in [Chien, Muscettola, et al., 1998].
5. Committed search places fewer requirements on the structure and complexity of auxiliary, special-purpose reasoning modules. In spacecraft commanding, these modules provide the planning system with information about spacecraft functions such as navigation, attitude control, power management, and thermal-constraint management. The simplest way for these modules to operate is to return a specific value for each query. For example, the planning system might need to ask the attitude-control module for a turn path that will safely turn the spacecraft from one pointing direction to another. The exact turn path can vary significantly, depending on its exact start time, because certain spacecraft areas must not turn toward certain celestial bodies (the camera must not face the sun, for example). This is further complicated in that the relative position of celestial bodies varies over time, particularly during a flyby. To fully exploit the power of a partially constrained plan representation, the planning system must use a more abstract information description: instead of a single turn path, the planning system needs a range of start times and path durations that stay within prespecified limits. However, building such abstractions can be costly and might not give the approximation level necessary to guarantee planning-system consistency in every possible execution. The balance between additional abstraction effort and potential payoff in terms of reduced search and solution flexibility will vary, depending upon the domain.

HEURISTIC ALGORITHM PORTFOLIOS

Algorithm Portfolios are sets of algorithms designed to work synergistically to solve a problem [Huberman et al. 1997, Fukunaga 2000]. The algorithms in a portfolio are synergistic in that the likelihood of success of the algorithms in the portfolio are negatively correlated. Thus, the probability that at least one of the algorithms in the portfolio will succeed is greater than the sum of the probabilities that independently one of the algorithms will succeed. Consider a simple two algorithm portfolio, consisting of algorithm A and algorithm B with the solution probability indicated below. Let P(X) indicate the probability that algorithm X will solve a problem instance given certain computational resources and P(~Y|X) denote the probability that algorithm X will solve a problem instance given that algorithm Y is not able to solve the same instance , each given the same computational resources. If X and Y are complementary, P(~Y|X) > P(X). However, P(X or Y) = P(Y) + P(~Y|X), hence in this case P(X or Y) > P(X) + P(Y). This example is depicted in the three graphs below. However, in the third graph, the P(A or B) presumes both algorithms running concurrently (e.g. using twice as much CPU per time unit).



Generally speaking, attaining an optimal portfolio mix requires reasoning about the marginal expected utility gain from allocating computational resources to the possible algorithms in the portfolio. For different deadlines, different portfolios may be optimal. Additionally, for many stochastic algorithms, restarts or multiple runs of the same algorithm may be beneficial to a portfolio (e.g. for heavy-tailed distributions [Gomes & Selman 1997]). Finally, algorithms can be combined within a single run to enable multiple algorithms to synergistically solve a single problem instance (approaching multi-strategy cooperative problem-solving).

ASPEN currently utilizes stochastic combination of a portfolio of choice heuristics to make choices at each step in an iterative repair search [Rabideau et al. 1999]. In this approach, at each choice point (e.g., conflict selection; resolution method selection - move, add, delete; activity selection), stochastic choice is made from a user-defined portfolio of heuristics. This technique allows the search to escape from looping and local minima and has enabled solution of a wide range of real-world planning and scheduling problems [Chien et al. 2000]. However, this approach suffers from lack of a strategic focused view of how to fix or improve the schedule.

We are currently investigating methods of addressing this shortcoming by developing: 1. deeper portfolio combination methods to allow for more focused algorithm application; and 2. specialized problem resolution methods for space application related problem classes (inspired by bottleneck centered resolution for production scheduling). Indeed, our general view of algorithm portfolios intentionally includes previous work in *multi-perspective scheduling* (e.g., OPIS [Smith 1994]) and *asynchronous teams* (A-teams) [Murthy et al. 1999]. Previous work in under the description of algorithm portfolios has generally used a fixed allocation of algorithms that does not change as a function of problem features, solution history, or feedback from algorithm performance. We propose to enable more powerful algorithm portfolios by relaxing each of these assumptions.

Problem features can be used to drive algorithm selection. For example, the OPIS system [Smith 1994] can be viewed as using problem features gleaned from bottleneck analysis (conflict duration, conflict size, idle time, ...) to select among a portfolio of specialized schedule repair algorithms. Asynchronous teams (A-Teams) can also be viewed as a portfolio of specialized algorithms that cooperate to solve problems (such as scheduling). This approach has been extremely successful in applications such as paper production management [Murthy et al. 1999] as well as shoe production planning, and sequence intensive planning [Crawford 2000]. We are developing of portfolios of specialized scheduling algorithms designed for space applications. This research will include development of the individual specialized algorithms (see below) as well as research into effective means to combine and coordinate the individual methods.

Individual algorithm performance can also be considered an extremely informative problem feature for use in algorithm selection. For example, the Operations Mission Planner (OMP) [Biefeld and Cooper, 1991] used a memory of (recent) past scheduler actions (called process chronologies) to drive algorithm choice. We are investigating algorithm portfolio selection policies based on past choices and heuristic application as well as the results of these choices. By enabling this decision process to be history-based we will be able to represent of powerful search methods

such as Limited Discrepancy Search (LDS) [Harvey & Ginsberg, 1995, Korf 1996] in the ASPEN search framework (LDS can be viewed as a portfolio that imposes a limitation on the number of selections of a random choice algorithm in the path from root to leaf (or sub-root to leaf)).

APPLICATIONS OF CASPER

The Three Corner Sat (3CS) University Nanosat mission will be using the CASPER onboard planning software integrated with the SCL ground and flight execution software [Chien et al. 2001a]. The 3CS mission is scheduled for launch in late 2002. 3CS will use onboard science data validation, replanning, robust execution, and multiple model-based anomaly detection. The 3CS mission is considerably less complex than Techsat-21 but still represents an important step in the integration and flight of onboard autonomy software. CASPER will fly on the Techsat-21 mission and will demonstrate an integrated autonomous mission using onboard science analysis, replanning, robust execution, model-based estimation and control, and formation flying. This demonstration is called the Autonomous Sciencecraft Constellation (ASC) [Chien et al. 2001b]. ASC will perform intelligent science data selection that will lead to a reduction in data downlink. In addition, the ASC experiment will increase science return through autonomous retargeting. CASPER is also being used in a number of research prototypes in applications ranging from autonomous rovers to deep space communications station automation [Knight et al. 2001]

CONCLUSIONS

This paper has described the local search framework most commonly used in the CASPER continuous planning system. This local search framework favors local, committed search because it facilitates incremental replanning as appropriate for an embedded system. The committed search framework also reduces efforts to interface automated planning software to other specialized reasoning modules (such as navigation, maneuver, path-planning, power management, or communications analysis software).This search framework has proven to be flexible and robust in a number of autonomous control applications including spacecraft operation, rovers, ground communications stations, and unpiloted aerial vehicles.

Acknowledgement

This work was performed at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration.

References

E. Biefeld and L. Cooper, "Bottleneck Identification Using Process Chronologies," *Proceedings of the 1991 International Joint Conference on Artificial Intelligence*, Sydney, Australia, 1991.

S. Chien, N. Muscettola, K. Rajan, B. Smith, G. Rabideau, "Automated Planning and Scheduling for Goal-based Autonomous Spacecraft," *IEEE Intelligent Systems*, September/October 1998, pp. 50-55.

S. Chien, R. Knight, A. Stechert, R. Sherwood, and G. Rabideau, "Using Iterative Repair to Improve Responsiveness of Planning and Scheduling," *Proceedings of the Fifth International Conference on Artificial Intelligence Planning and Scheduling*, Breckenridge, CO. April, 2000.

S. Chien, G. Rabideau, R. Knight, R. Sherwood, B. Engelhardt, D. Mutz, T. Estlin, B. Smith, F. Fisher, T. Barrett, G. Stebbins, D. Tran, "ASPEN - Automating Space Mission Operations using Automated Planning and Scheduling," *SpaceOps*, Toulouse, France, June 2000.

S. Chien, R. Knight, A. Stechert, R. Sherwood, and G. Rabideau, "Using Iterative Repair to Improve Responsiveness of Planning and Scheduling," *Proc Fifth International Conference on Artificial Intelligence Planning and Scheduling,* Breckenridge, CO, April 2000.

S. Chien, B. Engelhardt, R. Knight, G. Rabideau, R. Sherwood, E. Hansen, A. Ortiviz, C. Wilklow, S. Wichman " Onboard Autonomy on the Three Corner Sat Mission," *Intl Symposium on Artificial Intelligence Robotics and Automation in Space*, Montreal, Canada, June 2001

J. Crawford, Personal Communication, April 2000.

A. Fukunaga, "Genetic Algorithm Portfolios," *International Conf on Evolutionary Computation*, 2000.

C. Gomes, B. Selman, "Algorithm Portfolio Design: Theory vs. Practice." *Procedings of UAI-97*, RI. 1997.

W. Harvey and M. Ginsberg, "Limited discrepancy search," *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 607--613, 1995

B. Huberman, R. Lukose, T. Hogg, "An Economic Approach to Hard Computational Problems, *Science* v. 275, 3 January 1997, pp. 51-54.

A. Jonsson, P. Morris, N. Muscettola, K. Rajan, and B. Smith, "Planning in Interplanetary Space: Theory and Practice," *Procs 5th Int Conference on Artificial Intelligence Planning Systems, Breckenridge*, CO, April 2000.

R. Knight, G. Rabideau, S. Chien, "Computing Valid Intervals for Collections of Activities with Shared States and Resources," *Proc 5th Intl Conf Artificial Intelligence Planning and Scheduling*, Breckenridge, CO. April, 2000.

R. Knight, F. Fisher, T. Estlin, B. Engelhardt, S. Chien , "Balancing Deliberation and Reaction, Planning and Execution for Space Robotic Applications, "*Proceedings International Conference on Robotic Systems*, Wailea, HI, October 2001.

R. Korf, Improved Limited Discrepancy Search, *Proceedings of the 13th National Conference on Artificial Intelligence*, pp. 286-291, 1996.

S. Murthy, R. Akkiraju, R. Goodwin, P. Keskinocak, J. Rachlin, F. Wu ,S. Kumaran, J. Yeh, R. Fuhrer, A. Aggarwal, M. Sturzenbecker, R. Jayaraman, B. Daigle, "Cooperative multiobjective decision support for the paper industry," *Interfaces* (29) 5 pp.5-30, 1999.

G. Rabideau, R. Knight, S. Chien, A. Fukunaga, A. Govindjee, "Iterative Repair Planning for Spacecraft Operations in the ASPEN System," *Intl Symp on Artificial Intelligence Robotics and Automation in Space*, Noordwijk, The Netherlands, June 1999.

R. Sherwood et al., ASPEN Users Guide, http://aspen.jpl.nasa.gov

S. Smith, "OPIS: An Architecture and Methodology for Reactive Scheduling," in *Intelligent Scheduling*, Morgan Kaufman, 1994.

M. Zweben, B. Daun, E. Davis, and M. Deale, "Scheduling and Rescheduling with Iterative Repair," in *Intelligent Scheduling*, Morgan Kaufman, San Francisco, 1994.